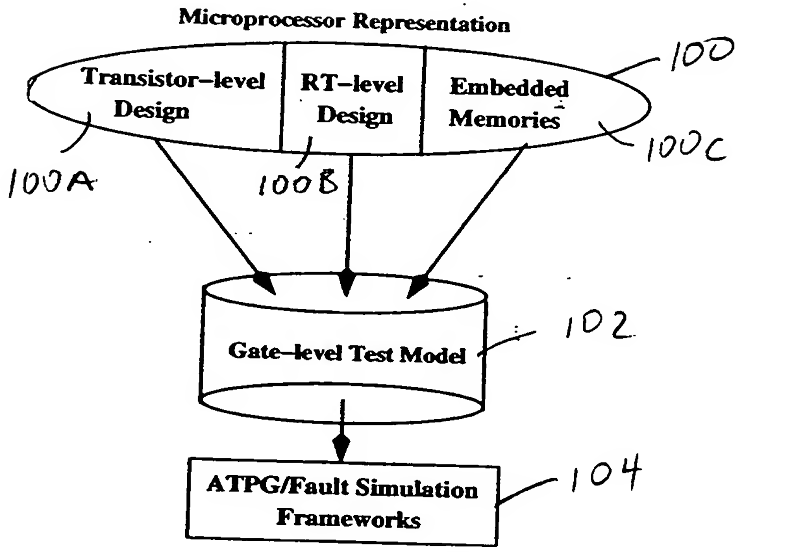


(Prior Art)

Fig. 1

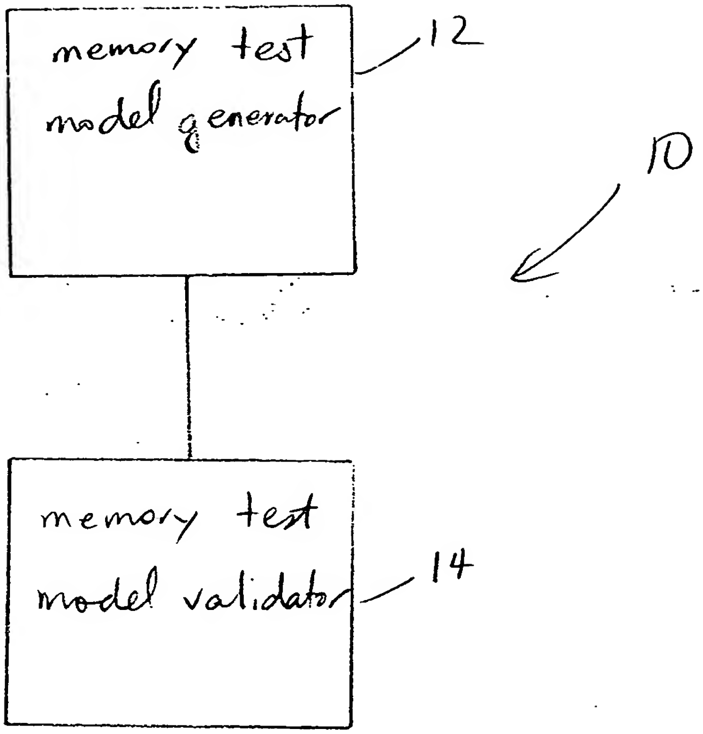

memory test model generator
12
10
memory test model validator
14

Fig. 2

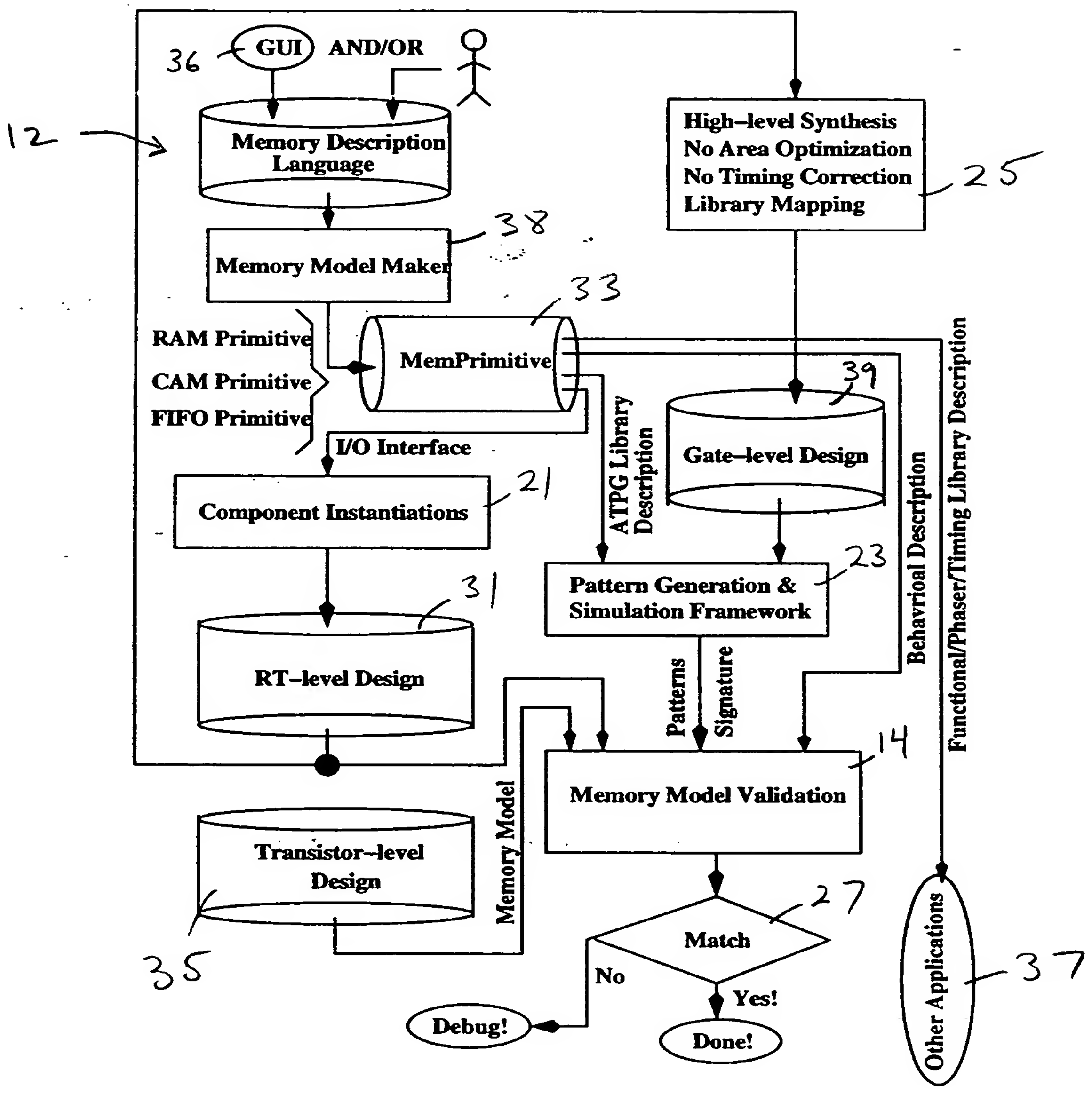

36
GUI
AND/OR
12
Memory Description Language
38
Memory Model Maker
33
RAM Primitive
CAM Primitive
FIFO Primitive
MemPrimitive
I/O Interface
21
Component Instantiations
31
RT-level Design
High-level Synthesis
No Area Optimization
No Timing Correction
Library Mapping
25
39
Gate-level Design
ATPG Library Description
23
Pattern Generation & Simulation Framework
Patterns
Signature
Behavrioal Description
Functional/Phaser/Timing Library Description
14
Memory Model Validation
Memory Model
Transistor-level Design
35
27
Match
No
Yes!
Debug!
Done!
Other Applications
37

Fig. 3

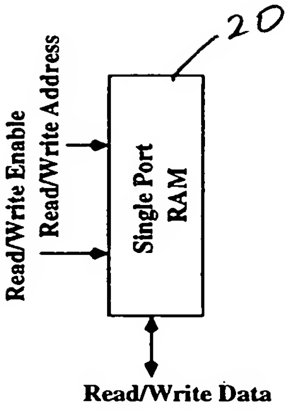


Fig. 4A

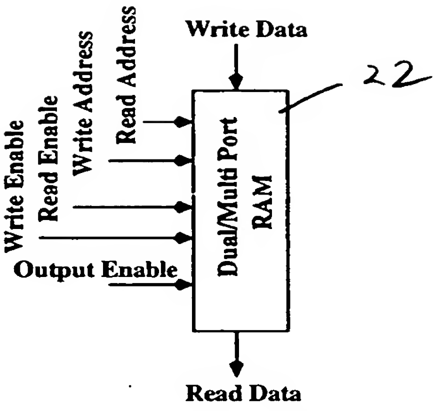


Fig. 4B

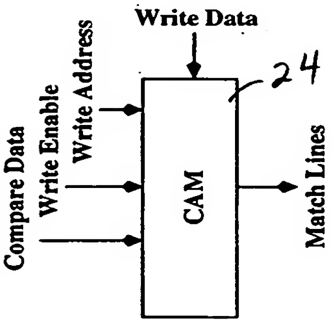


Fig. 4C

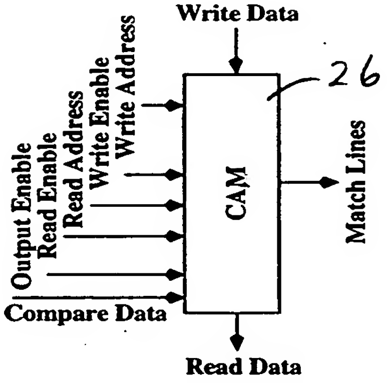


Fig. 4D

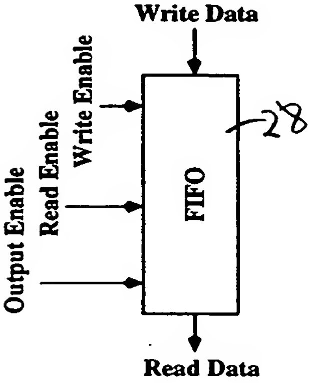


Fig. 4E

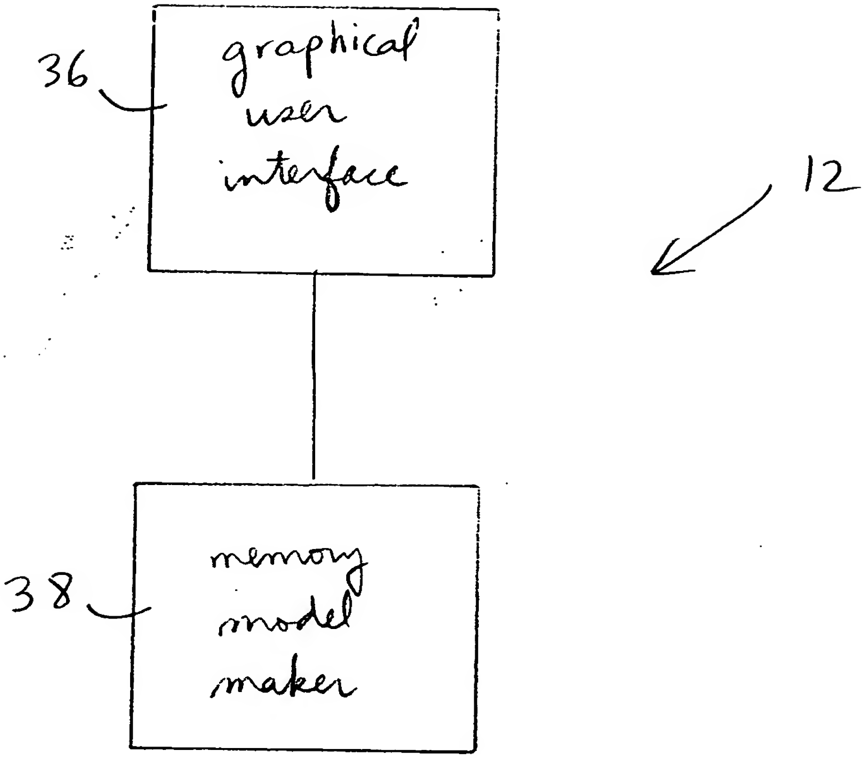
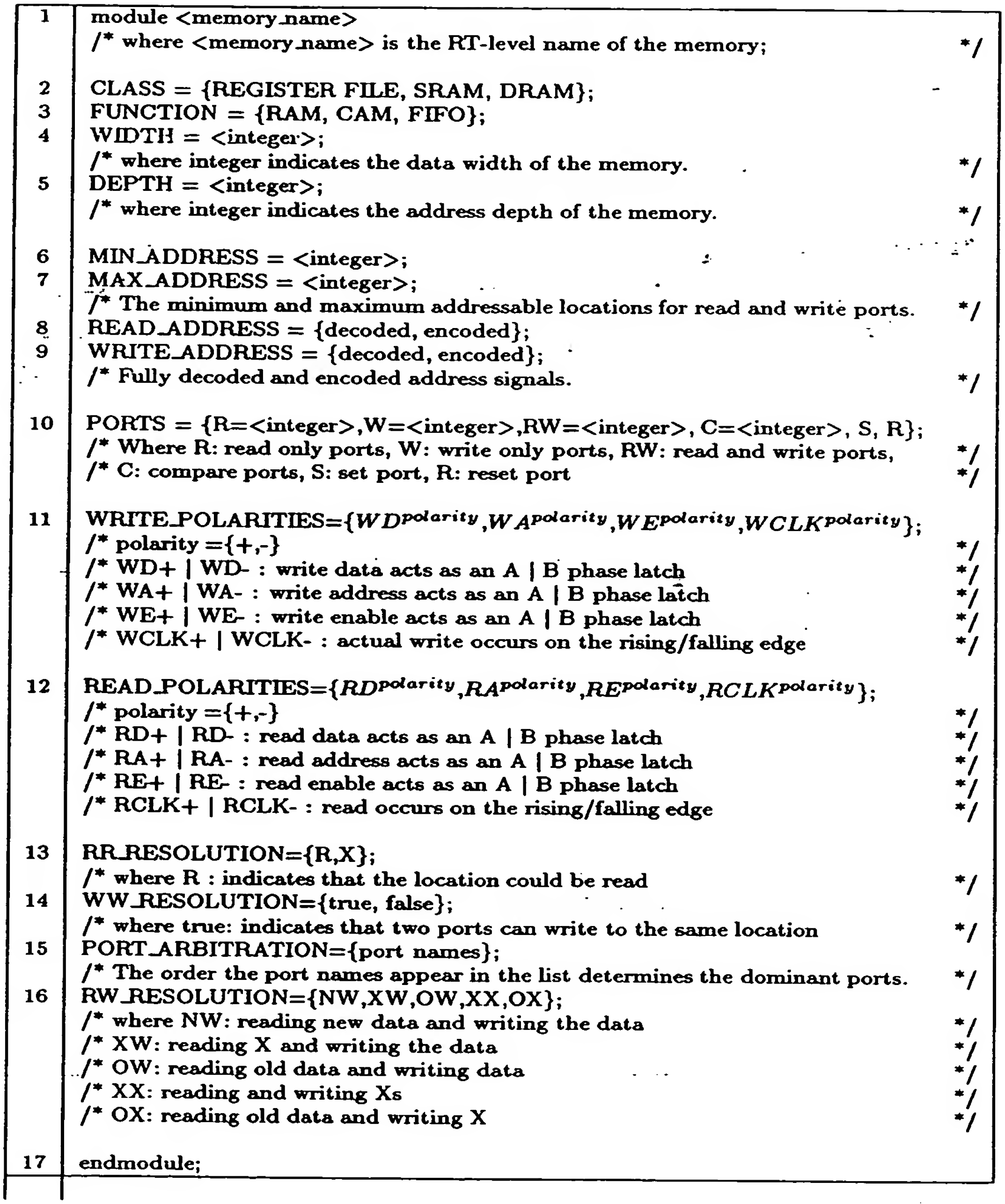


Fig. 5



```
1   module <memory_name>
    /* where <memory_name> is the RT-level name of the memory;                       */

2   CLASS = {REGISTER FILE, SRAM, DRAM};
3   FUNCTION = {RAM, CAM, FIFO};
4   WIDTH = <integer>;
    /* where integer indicates the data width of the memory.                         */
5   DEPTH = <integer>;
    /* where integer indicates the address depth of the memory.                      */

6   MIN_ADDRESS = <integer>;
7   MAX_ADDRESS = <integer>;
    /* The minimum and maximum addressable locations for read and write ports.       */
8   READ_ADDRESS = {decoded, encoded};
9   WRITE_ADDRESS = {decoded, encoded};
    /* Fully decoded and encoded address signals.                                     */

10  PORTS = {R=<integer>,W=<integer>,RW=<integer>, C=<integer>, S, R};
    /* Where R: read only ports, W: write only ports, RW: read and write ports,       */
    /* C: compare ports, S: set port, R: reset port                                   */

11  WRITE_POLARITIES={$WD^{polarity}$,$WA^{polarity}$,$WE^{polarity}$,$WCLK^{polarity}$};
    /* polarity ={+,-}                                                                */
    /* WD+ | WD- : write data acts as an A | B phase latch                            */
    /* WA+ | WA- : write address acts as an A | B phase latch                         */
    /* WE+ | WE- : write enable acts as an A | B phase latch                          */
    /* WCLK+ | WCLK- : actual write occurs on the rising/falling edge                 */

12  READ_POLARITIES={$RD^{polarity}$,$RA^{polarity}$,$RE^{polarity}$,$RCLK^{polarity}$};
    /* polarity ={+,-}                                                                */
    /* RD+ | RD- : read data acts as an A | B phase latch                             */
    /* RA+ | RA- : read address acts as an A | B phase latch                          */
    /* RE+ | RE- : read enable acts as an A | B phase latch                           */
    /* RCLK+ | RCLK- : read occurs on the rising/falling edge                         */

13  RR_RESOLUTION={R,X};
    /* where R : indicates that the location could be read                            */
14  WW_RESOLUTION={true, false};
    /* where true: indicates that two ports can write to the same location            */
15  PORT_ARBITRATION={port names};
    /* The order the port names appear in the list determines the dominant ports.     */
16  RW_RESOLUTION={NW,XW,OW,XX,OX};
    /* where NW: reading new data and writing the data                                */
    /* XW: reading X and writing the data                                             */
    /* OW: reading old data and writing data                                          */
    /* XX: reading and writing Xs                                                     */
    /* OX: reading old data and writing X                                             */

17  endmodule;
```

Fig. 6

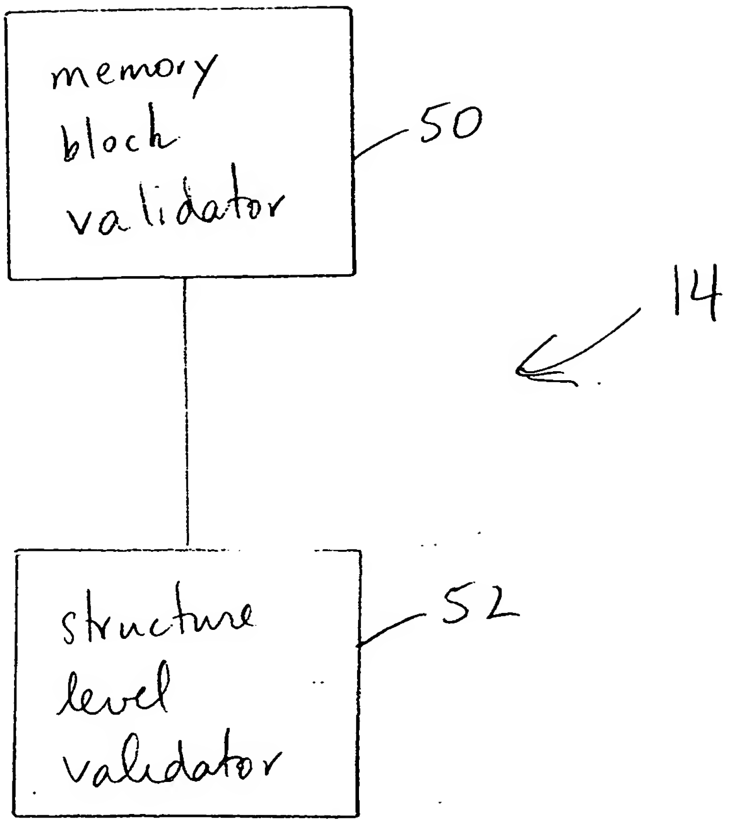


Fig. 7

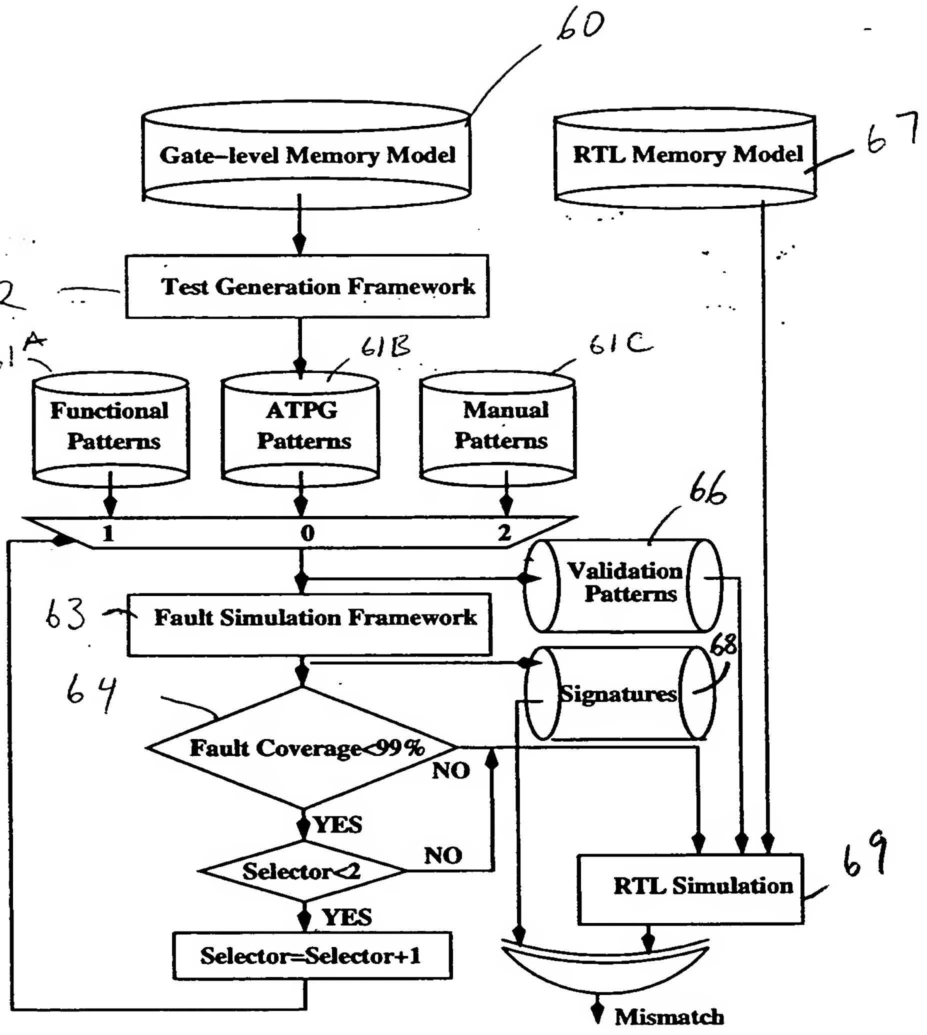

60
Gate–level Memory Model
RTL Memory Model
67
62
Test Generation Framework
61A
61B
61C
Functional Patterns
ATPG Patterns
Manual Patterns
1
0
2
66
Validation Patterns
63
Fault Simulation Framework
68
Signatures
64
Fault Coverage<99%
NO
YES
Selector<2
NO
YES
Selector=Selector+1
RTL Simulation
69
Mismatch

Fig. 8